AI for fun and serious puzzles

Bridging symbolic and numerical AI to learn how to solve Sudokus or design new proteins

Thomas Schiex
Université Fédérale de Toulouse, INRÆ MIAT, ANITI, Toulouse

November 2020

GdR IA - BIOSS seminar

Constraint network (X, C) — Feasibility biased

- a sequence X of discrete variables x_i, domain D_i

Constraint network (X, C) — Feasibility biased

- a sequence X of discrete variables x_i, domain D_i
- a set C of constraints

Constraint network (X, C) — **Feasibility biased**

- a sequence X of discrete variables x_i, domain D_i
- a set C of constraints
- $c_S \in C$ involves variables in $S \subseteq X$ and is a boolean function $\prod_{i \in S} D_i \rightarrow \{t, f\}$

Constraint network (X, C) — Feasibility biased

- a sequence X of discrete variables x_i, domain D_i
- a set C of constraints
- $c_S \in C$ involves variables in $S \subseteq X$ and is a boolean function $\prod_{i \in S} D_i \rightarrow \{t, f\}$
- a solution is an assignment of X that satisfies all constraints (NP-complete)

Symbolic NP-complete AI: SAT, CSP, CP

Constraint network (X, C) — Feasibility biased

- a sequence X of discrete variables x_i, domain D_i
- a set C of constraints
- $c_S \in C$ involves variables in $S \subseteq X$ and is a boolean function $\prod_{i \in S} D_i \rightarrow \{t, f\}$
- a solution is an assignment of X that satisfies all constraints (NP-complete)

SAT and CP

- Algorithms to find a solution (Backtrack, unit/constraint propagation, clause learning...)
- CP: predefined constraints (AllDifferent,...)

A bit of bragging...

NP-complete: can express all NP-complete problems

- the logical puzzles you like (Sudoku, Nonograms...)
- or not (configuration, scheduling, test pattern generation...)
- robot planning
- digital circuit verification (Bounded Model Checking)
- or software verification (FOL, grounding, abstraction)
- recently proved the “Pythagorean triple” conjecture (200TB proof)[10]

A bit of bragging...

NP-complete: can express all NP-complete problems

- the logical puzzles you like (Sudoku, Nonograms...)
- or not (configuration, scheduling, test pattern generation...)
- robot planning
- digital circuit verification (Bounded Model Checking)
- or software verification (FOL, grounding, abstraction)
- recently proved the “Pythagorean triple” conjecture (200TB proof)[10]

NP-complete, so intractable

Standard argument for less realistic problem reformulation, heuristics or stochastic search

A bit of bragging...

NP-complete: can express all NP-complete problems

- the logical puzzles you like (Sudoku, Nonograms...)
- or not (configuration, scheduling, test pattern generation...) SIEMENS THALES
- robot planning (Rosetta-Philæ probe plan, CP, LAAS/Toulouse) cnes
- digital circuit verification (Bounded Model Checking) intel IBM
- or software verification (FOL, grounding, abstraction) cadence
- recently proved the "Pythagorean triple" conjecture (200TB proof)[10]

NP-complete, so intractable

Standard argument for less realistic problem reformulation, heuristics or stochastic search

Real SAT instances with millions of variables/clauses can be solved (with a proof)

Is it bio-compatible?

Biology

- Many discrete object ($\{A, T/U, G, C\}$, amino acids, genes, alleles, enzymes...)
- Lots of experimental (noisy) data
- Biology has few “exact” rules

Is it bio-compatible?

Biology

- Many discrete object ($\{A, T/U, G, C\}$, amino acids, genes, alleles, enzymes...)
- Lots of experimental (noisy) data
- Biology has few “exact” rules

Shift from boolean function to numerical cost functions

From boolean functions to cost functions

Cost function network (X, W) — Homogeneous feasibility and criteria

- a sequence X of discrete variables x_i, domain D_i

From boolean functions to cost functions

Cost function network (X, W) — Homogeneous feasibility and criteria

- a sequence X of discrete variables x_i, domain D_i
- a set W of cost functions

From boolean functions to cost functions

Cost function network (X, W) — Homogeneous feasibility and criteria

- a sequence X of discrete variables x_i, domain D_i
- a set W of cost functions
- $w_S \in W$ is a numerical function $\prod_{i \in S} D_i$ (possibly infinite costs)

From boolean functions to cost functions

Cost function network (X, W) — Homogeneous feasibility and criteria

- a sequence X of discrete variables x_i, domain D_i
- a set W of cost functions
- $w_S \in W$ is a numerical function $\prod_{i \in S} D_i$ (possibly infinite costs)
- a solution optimizes the joint cost $W(X) = \sum_{w_S \in W} w_S(X[S])$ (WCSP, NP-complete)

From boolean functions to cost functions

Cost function network (X, W) — Homogeneous feasibility and criteria

- a sequence X of discrete variables x_i, domain D_i
- a set W of cost functions
- $w_S \in W$ is a numerical function $\prod_{i \in S} D_i$ (possibly infinite costs)
- a solution optimizes the joint cost $W(X) = \sum_{w_S \in W} w_S(X[S])$ (WCSP, NP-complete)

Generalizes CP: a constraint is a cost function that maps to $\{0, \infty\}$

From boolean functions to cost functions

Cost function network (X, W) — Homogeneous feasibility and criteria

- a sequence X of discrete variables x_i, domain D_i
- a set W of cost functions
- $w_S \in W$ is a numerical function $\prod_{i \in S} D_i$ (possibly infinite costs)
- a solution optimizes the joint cost $W(X) = \sum_{w_S \in W} w_S(X[S])$ (WCSP, NP-complete)

Generalizes CP: a constraint is a cost function that maps to $\{0, \infty\}$

Solvers: daoopt, toulbar2, MaxHS (MaxSAT)...

- Algorithms to find optimal solution (Branch and bound, cost function propagation)
- Predefined cost functions (Weighted All-Different,...)

Example: MAXCUT with hard edges

Graph $G = (V, E)$ with edge weight function w

- A boolean variable x_i per vertex $i \in V$
- A cost function per edge $e = (i, j) \in E$: $w_{ij} = w(i, j) \times \mathbb{1}[x_i \neq x_j]$
- Hard edges: constraints with costs 0 or $-\infty$ (when $x_i \neq x_j$)

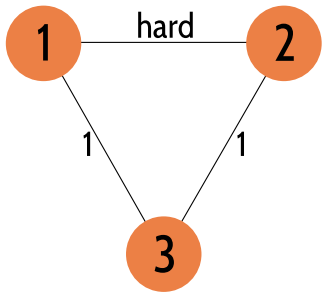
Example: MAXCUT with hard edges

Graph $G = (V, E)$ with edge weight function w

- A boolean variable x_i per vertex $i \in V$
- A cost function per edge $e = (i, j) \in E : w_{ij} = w(i, j) \times \mathbb{1}[x_i \neq x_j]$
- Hard edges: constraints with costs 0 or $-\infty$ (when $x_i \neq x_j$)

3-clique

- vertices $\{1, 2, 3\}$
- cut weight 1
- edge $(1, 2)$ hard.



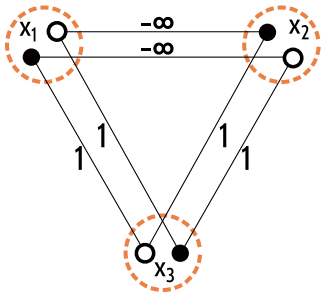
Example: MAXCUT with hard edges

Graph $G = (V, E)$ with edge weight function w

- A boolean variable x_i per vertex $i \in V$
- A cost function per edge $e = (i, j) \in E : w_{ij} = w(i, j) \times \mathbb{1}[x_i \neq x_j]$
- Hard edges: constraints with costs 0 or $-\infty$ (when $x_i \neq x_j$)

3-clique

- vertices $\{1, 2, 3\}$
- cut weight 1
- edge $(1, 2)$ hard.



toulbar2 input file (Python API too)

MAXCUT on a 3-clique with hard edge

```
{
 "problem" :{"name": "MaxCut", "mustbe": ">0.0"},
 "variables": {"x1": ["l","r"], "x2": ["l","r"], "x3": ["l","r"]},
 "functions": {
     "cut12": {"scope": ["x1","x2"], "costs": [0,-100,-100,0]},
     "cut13": {"scope": ["x1","x3"], "costs": [0,1,1,0]},
     "cut23": {"scope": ["x2","x3"], "costs": [0,1,1,0]}
  }
}
```

Solving the Weighted Constraint Satisfaction Problem

Extensions of boolean reasoning to costs[21]

- Massive local reasoning provides (LP-related) lower bounds[5,6]
- Backtrack turned into (HBFS) Branch and Bound[1]
- Clever adaptive heuristics[3,12]
- On the fly variable elimination,[11] dominance,[2,6]...
- Graph-structure (treewidth/treedepth) aware[7]
- Guaranteed hybrid local search (VNS)[15]

Toulbar2 (github.com/toulbar2/toulbar2)

- Open Source C++ CFN solver (github.com/toulbar2/toulbar2)
- Won the UAI/Pascal “Approximate Probabilistic Inference Challenge” twice (and more)

Probabilistic Inference?

A CFN is a "Graphical Model"[8]

- Defines a joint non negative integer function on many variables
- Very close to Markov Random Field (Potts models): defines a probability distribution
- The WCSP is almost the "Maximum a Posteriori" or MAP/MRF problem

Probabilistic Inference?

A CFN is a "Graphical Model"[8]

- Defines a joint non negative integer function on many variables
- Very close to Markov Random Field (Potts models): defines a probability distribution
- The WCSP is almost the "Maximum a Posteriori" or MAP/MRF problem

Cost/energy and probability

$$P(X) \propto e^{-W(X)}$$

Probabilistic Inference?

A CFN is a "Graphical Model"[8]

- Defines a joint non negative integer function on many variables
- Very close to Markov Random Field (Potts models): defines a probability distribution
- The WCSP is almost the "Maximum a Posteriori" or MAP/MRF problem

Cost/energy and probability

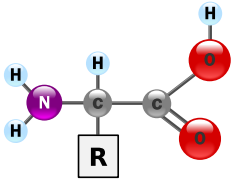
$$P(X) \propto e^{-W(X)}$$

Toulbar2 solves MAP/MRF+constraints using automated reasoning algorithms Uncertainty Ok

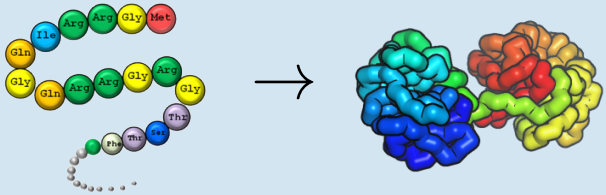
Proteins

Most active molecules of life

Sequence of “amino-acids”, each chosen among 20 natural ones



Folding

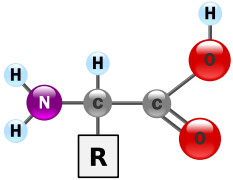


Transporter, binder, regulator, motor, catalyst...
Hemoglobine, TAL effector, ATPase, dehydrogenases...

Protein Design

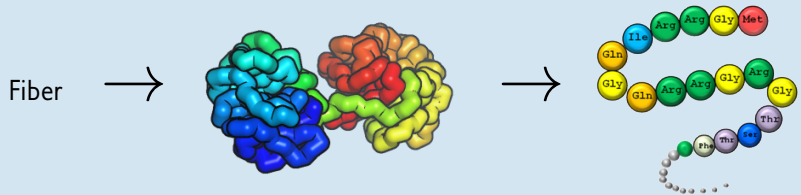
Most active molecules of life

Sequence of “amino-acids”, each chosen among 20 natural ones



Inverse folding

Fiber



Transporter, binder, regulator, motor, catalyst...
Hemoglobine, TAL effector, ATPase, dehydrogenases...

Why should we design new proteins

Eco-friendly chemical/structural nano-agents

- New components for nanotechnologies
- New catalysts (environment, biofuels, food and feed, cosmetics...),
- New drugs for medicine (antibodies,...)

Why should we design new proteins

Eco-friendly chemical/structural nano-agents

- New components for nanotechnologies
- New catalysts (environment, biofuels, food and feed, cosmetics...),
- New drugs for medicine (antibodies,...)

20^n sequences! intractable for experimental techniques

Why should we design new proteins

Eco-friendly chemical/structural nano-agents

- New components for nanotechnologies
- New catalysts (environment, biofuels, food and feed, cosmetics...),
- New drugs for medicine (antibodies,...)

20^n sequences! intractable for experimental techniques

CPD: From bits to atoms — From information to functional matter

- mass 3d printing-like capacities at atomic level (bacterias, yeast, ...)
- structural and functional purposes (powerful origami)
- produced new folds,[9] catalysts,[19] nano-components[24]

Protein Design as the inverse of folding

Ingredients

- Full atom model of a protein backbone (assumed to be rigid)
- Full atom energy function (bonds, electrostatics, solvant, statistics...)
- Maximum stability $\equiv$ Minimum energy NP-hard[18]
- Catalog of all 20 amino acids in different conformations (≈ 400 overall)

Protein Design as the inverse of folding

Ingredients

- Full atom model of a protein backbone (assumed to be rigid)
- Full atom energy function (bonds, electrostatics, solvant, statistics...)
- Maximum stability $\equiv$ Minimum energy NP-hard[18]
- Catalog of all 20 amino acids in different conformations (≈ 400 overall)

As a cost function network

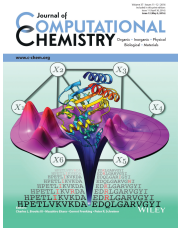
- One variable per position in the protein sequence
- Domain: catalog of amino acids conformations (rotamers)
- Function: decomposed pairwise energy
- Search space has size $\approx 400^n$

Exact vs. Stochastic search

Large input (> 1GB) — **NP-hard problem**

Toulbar2 is able to...

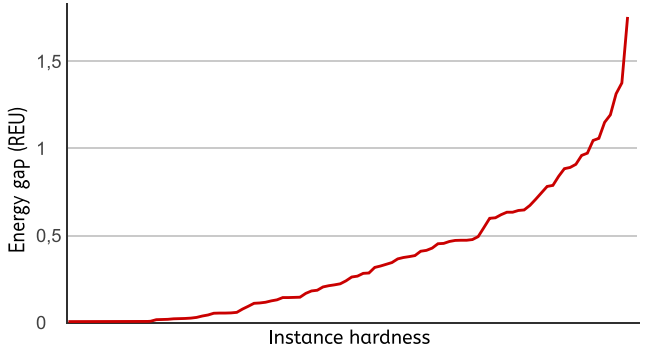
- provide a proven minimum energy solution
- exhaustively enumerate sequences close to it
- in spaces of size $> 10^{400}$



A highly tuned Monte Carlo Simulated Annealer increasingly fails to find the optimal sequence[a]

[a] David Simoncini et al. "Guaranteed Discrete Energy Optimization on Large Protein Design Problems". In: *Journal of Chemical Theory and Computation* 11.12 (2015), pp. 5980–5989. DOI: 10.1021/acs.jctc.5b00594.

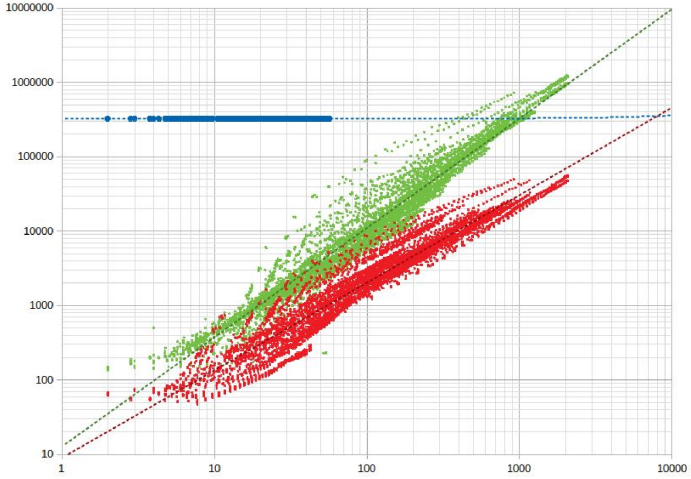
Unbounded error



Asymptote: **Size matters!**

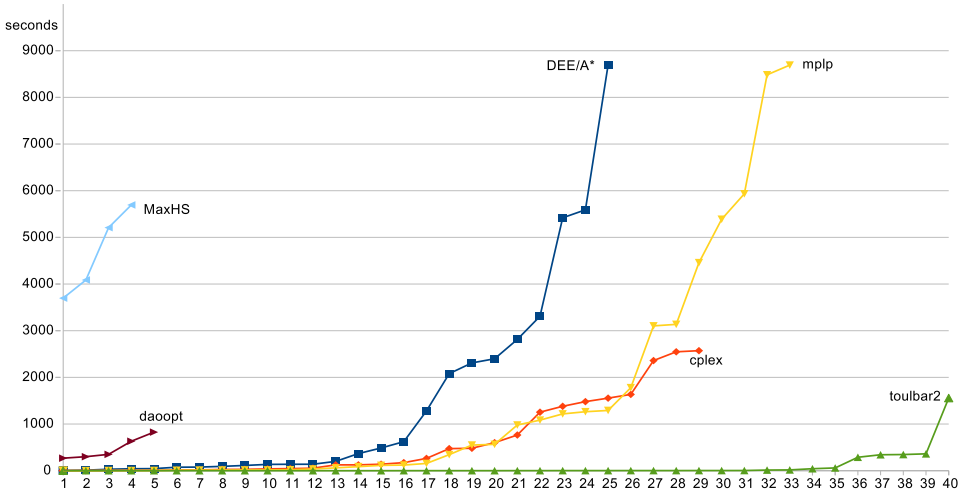
Asymptotic convergence can be arbitrarily slow...

Quantum computing (DWave),Toulbar2 & SA[13]



DWave approximations		*kcal/mol*
gap > 1.16, 90% of the time	> 4.35, 50% of the time	> 8.45, 10% of the time

Toulbar2 vs. CPLEX, MaxHS...(real instances)

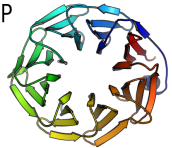


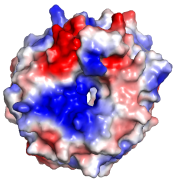
# of instances solved (*X*) within a per instance cpu-time limit (*Y*)

From bits to atoms (col. A. Vœt (KU Leuven), D. Simoncini[14])

C8 pseudo-symetric 2OVP symmetrized into a nano-component

2OVP

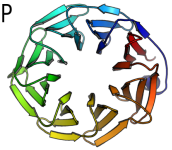




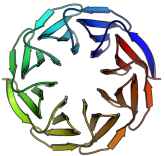
C8 pseudo-symetric 2OVP symmetrized into a nano-component

-  Tako: (R)evolution + Rosetta/talaris14 8 fold

2OVP



Tako

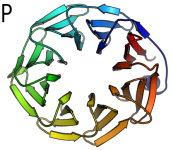


From bits to atoms (col. A. Vœt (KU Leuven), D. Simoncini[14])

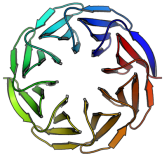
C8 pseudo-symetric 2OVP symmetrized into a nano-component

-  Tako: (R)evolution + Rosetta/talaris14 — 8 fold
-  Ika: toulbar2 + talaris14 — 4 fold

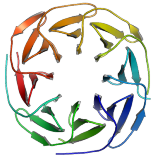
2OVP



Tako



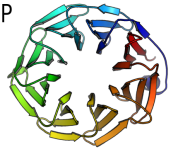
Ika



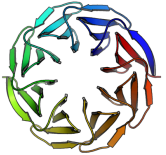
C8 pseudo-symetric 2OVP symmetrized into a nano-component

-  Ika: toulbar2 + talaris14 4 fold

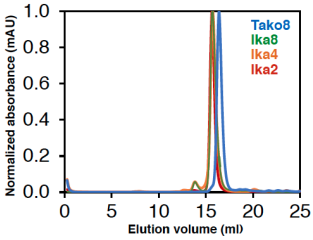
2OVP

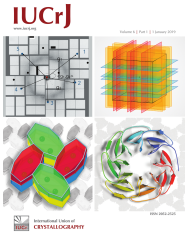


Tako



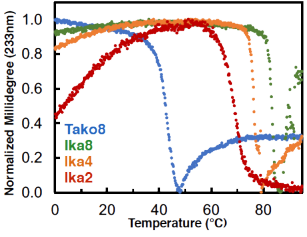
Ika

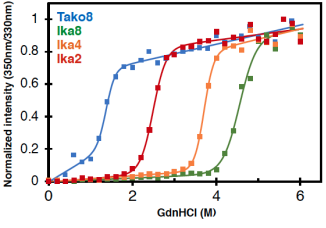




Assemble as 8-bladed propeller

- Ika* more stable than Tako8
- Temperature
- Chemical denaturation





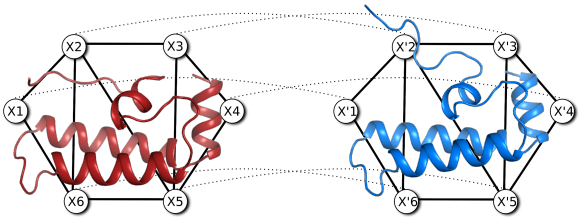
Multistate design accounts for protein flexibility

Optimize energy on several backbones

- Only desirable backbones (flexibility, switches,...): decision NP-complete
- Undesirable backbones (specificity, orthogonal design): Σ_2^p-complete

Positive multistate design[25]

Combine multiple CFN-models, connect them by sequence identity constraints



Positive multi-state design

Tested on 10 NMR and 10 X-Ray+backrub ensembles of 4 states

- large problems: more than 300 variables, several hundreds of values
- can be solved to optimality in reasonable time on one CPU corrected
- “slow” exponential growth of cpu-time as a function of input size

Native sequence recovery

- Fraction of amino acids of an existing proteins recovered by design
- Increases by an average of 16% (NMR) or 8% (X-ray+backrub) over single state design
- Better protein design with no algorithm design

Energy function

Imperfect

- Approximations: solvent effect...
- Ignored: polarisability, expressability...
- Needs more information, extracted from *data*

Energy function

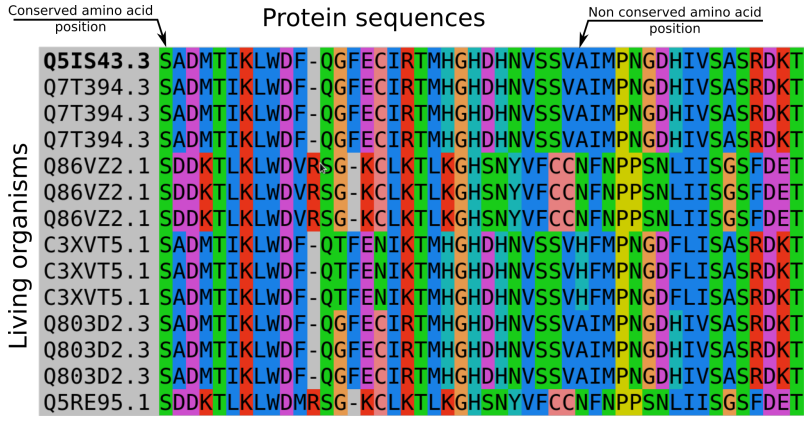
Imperfect

- Approximations: solvent effect...
- Ignored: polarisability, expressability...
- Needs more information, extracted from *data*

Evolutionary information

- Use similar proteins (homologs) from databases
- Multiple alignment: align similar regions of the sequences

A multiple alignment with conserved positions



Simple integration of information

- Force amino acid choice (constraint) at conserved positions.

Learning CFNs from data

Boltzman distribution connects probability and cost

$$P(X) \propto e^{-W(X)}$$

- Use (regularized) maximum-loglikelihood estimation
- Has a nice cost-domain interpretation

Learning in practice

Similar to contact-map prediction (CCMPred[22])

- We start from a complete pairwise CFN with unknown cost functions

Learning in practice

Similar to contact-map prediction (CCMPred[22])

- We start from a complete pairwise CFN with unknown cost functions
- We have a total of $d^2 \cdot \frac{n(n-1)}{2}$ parameters to learn $w_{ij}(\cdot,\cdot)$

Learning in practice

Similar to contact-map prediction (CCMPred[22])

- We start from a complete pairwise CFN with unknown cost functions
- We have a total of $d^2 \cdot \frac{n(n-1)}{2}$ parameters to learn $w_{ij}(\cdot,\cdot)$
- Let $\ell(D|w_{ij})$ be the log-probability of data D given the w_{ij}

Learning in practice

Similar to contact-map prediction (CCMPred[22])

- We start from a complete pairwise CFN with unknown cost functions
- We have a total of $d^2 \cdot \frac{n(n-1)}{2}$ parameters to learn $w_{ij}(\cdot,\cdot)$
- Let $\ell(D|w_{ij})$ be the log-probability of data D given the w_{ij}

Maximize $\ell(D|w_{ij}) - \lambda \cdot ||w_{ij}||$ L1, L1/L2 and L2

Learning in practice

Similar to contact-map prediction (CCMPred[22])

- We start from a complete pairwise CFN with unknown cost functions
- We have a total of $d^2 \cdot \frac{n(n-1)}{2}$ parameters to learn $w_{ij}(\cdot,\cdot)$
- Let $\ell(D|w_{ij})$ be the log-probability of data D given the w_{ij}

Maximize $\ell(D|w_{ij}) - \lambda \cdot ||w_{ij}||$ L1, L1/L2 and L2

Efficient convex optimization based implementation

- Uses the Alternating Direction Multiplier Method[17]
- Approximate likelihood based (probabilistic input) + L1 (sparsity)
- Reasonably good performances, even in Python (numpy)

Designing a new nanobody scaffold

- Using Rosetta hybrid force-field and rotamer library
- Trying to satisfy several constraints (originality, resistance, composition...)
- Multi-state design[25]
- with MSA-extracted evolutionary preferences

In Protein design

Designing a new nanobody scaffold

- Using Rosetta hybrid force-field and rotamer library
- Trying to satisfy several constraints (originality, resistance, composition...)
- Multi-state design[25]
- with MSA-extracted evolutionary preferences

Limited experimental power

- Over 6 sequences designed without evolutionary information: 3 expressed
- Over 3 sequences designed with evolutionary information: 3 expressed
- Much more power in a recent Science paper[20]

But much more general (sparsity crucial)[4]

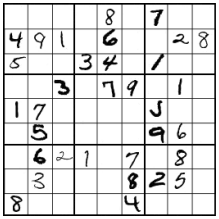
Can learn to play the Sudoku from examples

- Less examples, faster, better than RRN[16] or SAT-Net[26]
- With comparable or less biases
- and the ability to exploit perceptual "layers" output

But much more general (sparsity crucial)[4]

Can learn to play the Sudoku from examples

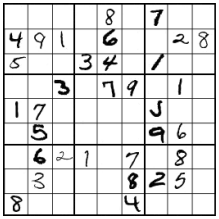
- Less examples, faster, better than RRN[16] or SAT-Net[26]
- With comparable or less biases
- and the ability to exploit perceptual "layers" output



But much more general (sparsity crucial)[4]

Can learn to play the Sudoku from examples

- Less examples, faster, better than RRN[16] or SAT-Net[26]
- With comparable or less biases
- and the ability to exploit perceptual "layers" output



Learn user preferences for car configuration assistant — Renault/Hélène Fargier

- 8,252 historical configurations of an higly combinatorial van (2^{74} configurations)
- extra configuration constraints
- best results (accuracy) obtained when combining both ML and CP

Combines ML (data) and Automated reasoning (logic)

Let's recap...

- Model the problem as a CFN (generalizes CP): knowledge

Combines ML (data) and Automated reasoning (logic)

Let's recap...

- Model the problem as a CFN (generalizes CP): knowledge
- Learn other CFNs from available data sets (convex optimization,...)

Combines ML (data) and Automated reasoning (logic)

Let's recap...

- Model the problem as a CFN (generalizes CP): knowledge
- Learn other CFNs from available data sets (convex optimization,...)
- Combine the models by scaling/adding/connecting them together

Combines ML (data) and Automated reasoning (logic)

Let's recap...

- Model the problem as a CFN (generalizes CP): knowledge
- Learn other CFNs from available data sets (convex optimization,...)
- Combine the models by scaling/adding/connecting them together
- Add further design constraints/preferences: desired properties

Open source

https://github.com/toulbar2/toulbar2
https://github.com/toulbar2/CFN-learn

Combines ML (data) and Automated reasoning (logic)

Let's recap...

- Model the problem as a CFN (generalizes CP): knowledge
- Learn other CFNs from available data sets (convex optimization,...)
- Combine the models by scaling/adding/connecting them together
- Add further design constraints/preferences: desired properties
- Solve them with toulbar2 : −) to get your new design

Open source

https://github.com/toulbar2/toulbar2
https://github.com/toulbar2/CFN-learn

Thanks

AI/toulbar2

S. de Givry (INRA)
G. Katsirelos (INRA)
M. Zytnicki (PhD, INRA)
D. Allouche (INRA)
M. Ruffini (INRA)
H. Nguyen (PhD, INRA)
C. Brouard (ML, INRA)
M. Cooper (IRIT, Toulouse)
J. Larrosa (UPC, Spain)
F. Heras (UPC, Spain)
M. Sanchez (Spain)
E. Rollon (UPC, Spain)
P. Meseguer (CSIC, Spain)
G. Verfaillie (ONERA, ret.)
JH. Lee (CU. Hong Kong)
C. Bessiere (LIMM, Montpellier)
JP. Métivier (GREYC, Cæn)
S. Loudni (GREYC, Cæn)
M. Fontaine (GREYC, Cæn),...

Protein Design

A. Vœt (KU Leuven)
A. Olichon (INSERM)
D. Simoncini (UFT, Toulouse)
S. Barbe (INSA, Toulouse)
J. Vucinic (INRA/INSA)
S. Traoré (PhD, CEA)
C. Viricel (PhD)
RosettaCommons (U. Washington)
W. Sheffler (U. Washington)
PyRosetta (U. John Hopkins)
B. Donald (U. North Carolina)
K. Roberts (U. North Carolina)
T. Simonson (Polytechnique)
J. Cortes (LAAS/CNRS),...

My apologies to those missing in these lists. Even imperfect lists seem better than simply no list

References I

[1] David Allouche et al. “Anytime Hybrid Best-First Search with Tree Decomposition for Weighted CSP”. In: *Principles and Practice of Constraint Programming*. Springer. 2015, pp. 12–29.

[2] David Allouche et al. “Computational protein design as an optimization problem”. In: *Artificial Intelligence* 212 (2014), pp. 59–79.

[3] Frédéric Boussemart et al. “Boosting systematic search by weighting constraints”. In: *ECAI*. Vol. 16. 2004, p. 146.

[4] Céline Brouard, Simon de Givry, and Thomas Schiex. “Pushing data into CP models using Graphical Model Learning and Solving”. In: *Principles and Practice of Constraint Programming–CP 2020*. Springer, 2020.

[5] Martin C Cooper et al. “Soft arc consistency revisited”. In: *Artificial Intelligence* 174.7 (2010), pp. 449–478.

[6] Simon De Givry, Steven D Prestwich, and Barry O’Sullivan. “Dead-end elimination for weighted CSP”. In: *Principles and Practice of Constraint Programming*. Springer. 2013, pp. 263–272.

[7] S. de Givry, T. Schiex, and G. Verfaillie. “Exploiting Tree Decomposition and Soft Local Consistency in Weighted CSP”. In: *Proc. of the National Conference on Artificial Intelligence, AAAI-2006*. 2006, pp. 22–27.

[8] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[9] Brian Kuhlman et al. “Design of a novel globular protein fold with atomic-level accuracy”. In: *science* 302.5649 (2003), pp. 1364–1368.

[10] Oliver Kullmann. “The Science of Brute Force”. In: *Communications of the ACM* (2017).

References II

[11] J. Larrosa. “Boosting search with variable elimination”. In: *Principles and Practice of Constraint Programming - CP 2000*. Vol. 1894. LNCS. Singapore, Sept. 2000, pp. 291–305.

[12] C. Lecoutre et al. “Reasoning from last conflict(s) in constraint programming”. In: *Artificial Intelligence* 173 (2009), pp. 1592, 1614.

[13] Vikram Khipple Mulligan et al. “Designing Peptides on a Quantum Computer”. In: *bioRxiv* (2019), p. 752485.

[14] Hiroki Noguchi et al. “Computational design of symmetrical eight-bladed β-propeller proteins”. In: *IUCrJ* 6.1 (2019).

[15] Abdelkader Ouali et al. “Variable neighborhood search for graphical model energy minimization”. In: *Artificial Intelligence* 278 (2020), p. 103194.

[16] Rasmus Palm, Ulrich Paquet, and Ole Winther. “Recurrent relational networks”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 3368–3378.

[17] Youngsuk Park et al. “Learning the network structure of heterogeneous data via pairwise exponential Markov random fields”. In: *Proceedings of machine learning research* 54 (2017), p. 1302.

[18] Niles A Pierce and Erik Winfree. “Protein design is NP-hard”. In: *Protein engineering* 15.10 (2002), pp. 779–782.

[19] Daniela Röthlisberger et al. “Kemp elimination catalysts by computational enzyme design”. In: *Nature* 453.7192 (2008), p. 190.

[20] William P Russ et al. “An evolution-based model for designing chorismate mutase enzymes”. In: *Science* 369.6502 (2020), pp. 440–445.

References III

[21] T. Schiex, H. Fargier, and G. Verfaillie. “Valued Constraint Satisfaction Problems: hard and easy problems”. In: *Proc. of the 14th IJCAI*. Montréal, Canada, Aug. 1995, pp. 631–637.

[22] Stefan Seemayer, Markus Gruber, and Johannes Söding. “CCMpred—fast and precise prediction of protein residue–residue contacts from correlated mutations”. In: *Bioinformatics* 30.21 (2014), pp. 3128–3130.

[23] David Simoncini et al. “Guaranteed Discrete Energy Optimization on Large Protein Design Problems”. In: *Journal of Chemical Theory and Computation* 11.12 (2015), pp. 5980–5989. DOI: 10.1021/acs.jctc.5b00594.

[24] Arnout RD Vœt et al. “Computational design of a self-assembling symmetrical β-propeller protein”. In: *Proceedings of the National Academy of Sciences* 111.42 (2014), pp. 15102–15107.

[25] Jelena Vucinic et al. “Positive multistate protein design”. In: *Bioinformatics* 36.1 (2020), pp. 122–130.

[26] Po-Wei Wang et al. “SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver”. In: *ICML’19 proceedings, arXiv preprint arXiv:1905.12149*. 2019.